

# Composable Security in the Bounded-Quantum-Storage Model

Stephanie Wehner \*  
 CWI, Amsterdam  
 wehner@cwi.nl

Jürg Wullschleger  
 University of Bristol  
 juerg@wulli.com

February 1, 2008

## Abstract

We present a simplified framework for proving sequential composability in the quantum setting. In particular, we give a new, simulation-based, definition for security in the bounded-quantum-storage model, and show that this definition allows for sequential composition of protocols. Damgård *et al.* (FOCS '05, CRYPTO '07) showed how to securely implement bit commitment and oblivious transfer in the bounded-quantum-storage model, where the adversary is only allowed to store a limited number of qubits. However, their security definitions did only apply to the standalone setting, and it was not clear if their protocols could be composed. Indeed, we first give a simple attack that shows that these protocols are *not* composable without a small refinement of the model. Finally, we prove the security of their randomized oblivious transfer protocol in our refined model. Secure implementations of oblivious transfer and bit commitment then follow easily by a (classical) reduction to randomized oblivious transfer.

## 1 Introduction

Secure two-party computation [46] allows two mutually distrustful players to jointly compute the value of a function without revealing more information about their inputs than can be inferred from the function value itself. In this context, the primitives known as bit commitment (BC) [6] and oblivious transfer (OT) [44, 33, 19] are of particular importance: *any* two-party computation can be implemented, provided these two primitives are available [20, 23, 15].

In bit commitment, the committer (Alice) secretly chooses a bit  $b$ , and commits herself to  $b$  by exchanging messages with the verifier (Bob). From the commitment alone, Bob should not be able to gain any information about  $b$ . Yet, when Alice later reveals  $b$  and opens the commitment by exchanging messages with Bob, he can verify whether Alice is truthful and had indeed committed herself to  $b$ . In oblivious transfer, the sender (Alice) chooses two bits  $x_0$  and  $x_1$ , the receiver (Bob) chooses a bit  $c$ . The protocol of oblivious transfer allows Bob to retrieve  $x_c$  in such a way that Alice cannot gain any information about  $c$ . At the same time, Alice can be ensured that Bob only retrieves  $x_c$  and no information about the other input bit  $x_{1-c}$ .

Unfortunately, BC and OT are impossible to implement securely without any additional assumptions, even in the quantum model [29, 26]. This result holds even in the presence of the so-called superselection rules [24]. Exact tradeoffs on how well we can implement BC in the quantum world can be found in [39]. To circumvent this problem (in both, the classical and the quantum

---

\*Supported by EU fifth framework project QAP IST 015848 and the NWO vici project 2004-2009.

case), we thus need to assume that the adversary is limited. In the classical case, one such limiting assumption is that the adversary is *computationally bounded*, i.e., he is restricted to a polynomial time computations (see e.g. [31, 19]). In the quantum model, it is also possible to securely implement both protocols provided that an adversary cannot measure more than a fixed number of qubits simultaneously [37]. Very weak forms of string commitments can also be obtained [7].

**The Bounded-Quantum-Storage Model.** Of particular interest to us is the bounded-storage model. Here, the adversary is bounded in *space* instead of time, i.e., she is only allowed to use a certain amount of storage space. Both OT and BC can be implemented in this model [9]. Yet, the security of a *classical* bounded-storage model [27, 9] is somewhat unsatisfactory: First, a dishonest player needs only quadratically more memory than the honest one. Second, as classical memory is very cheap, most of these protocols require a huge amount of communication in order to achieve reasonable bounds on the adversaries memory. In the quantum case, on the other hand, it is *very* difficult to store states even for a very short period of time. This leads to the protocol presented in [5, 14], which show how to implement BC and OT if the adversary is not able to store *any* qubits at all. In [17, 16], these ideas have been generalized in a very nice way to the *bounded-quantum-storage model*, where the adversary is computationally unbounded and allowed to have an unlimited amount of *classical* memory. However, he is only allowed a limited amount of *quantum* memory. The advantages over the classical bounded-storage model are two fold: First, given current day technology it is indeed very hard to store quantum states. Secondly, here the honest player does not require any quantum storage at all, making the protocol implementable using present day technology.

**Security Definitions and Composability.** Cryptographic protocols (especially protocols that implement very basic functionalities such as BC or OT) are almost never executed on their own. They are merely used as building blocks for larger, more complicated applications. However, it is not clear that the composition of secure protocols will remain secure. Formal security definitions for secure function evaluation have first been proposed in [30] and [2]. These definitions use the *simulation paradigm* invented in [21] to define zero-knowledge proofs of knowledge. In [10] it has been shown formally that these definitions imply that protocols can be composed *sequentially*. Sequential composition implies that protocols can be composed in an arbitrary way, as long as at any point in time exactly one protocol is running. All other protocols have to wait until that protocol stops. A stronger security definition called *universal composability* has been introduced in [11, 32, 1]. It guarantees that protocols can be securely composed in an arbitrary way (also concurrently) in any environment.

Simulation-based security requires that for any adversary attacking the real protocol there exists a simulator in the ideal setting, i.e. where the players only have black-box access to an ideal functionality, such that the environment cannot distinguish between the real and the ideal setting. To make the protocol sequentially composable, we have to allow the adversary to receive some *auxiliary input* from the environment, which could contain information from a previous run of the protocol, the larger application that the protocol is embedded in, or any other information that the environment might pass to the adversary in an attempt to distinguish between the real from the ideal setting. In the quantum case, this auxiliary input is an arbitrary quantum state, unknown to the adversary or the simulator. This presents us with two additional difficulties we do not encounter in the classical setting: First, the simulator cannot determine what this input

state is without disturbing the state, which could be detected by the environment. Second, the input state may be entangled with the environment. Based on earlier work in [42], a simulation-based framework for secure quantum multi-party computation has been presented in [38]. That framework offers sequential composability, however no composability theorem was presented there. Universal composability in the quantum world has been introduced in [4], and independently in [41]. Their framework is very powerful. But, due to their complexity, hard to apply.

In [40], it has been shown that classical protocols which have been proven to be universally composable using their *classical* definitions, are secure against *quantum* adversaries. This is result is very useful, as it allows us to use many classical protocols also in the quantum setting. Great care must be taken in the definition of security in the quantum setting: For example, the standard security definition for QKD based on accessible information does not imply compositability [25].

## 1.1 Contribution

In [16], protocols for OT and BC have been presented and shown to be secure against bounded quantum adversaries. However, the proofs only guarantee security in a standalone setting. Indeed, a very simple attack shows that they are not composable. The main contribution of this paper is to give a formal framework for security in the bounded-quantum-storage model, and to show that modified versions of these protocols are *sequentially composable*. Hence, they can be used as building blocks in other protocols.

**Proofs in [17, 16] do not imply Composability.** When considering composable security, we need to allow the adversary to receive some auxiliary quantum input. This has not been considered in the security definitions used in [17, 16]. When we allow this, we are faced with two major problems: First, in the security proof of [16], the receivers choice bit can only be extracted by the simulator if the distribution of the senders random string given the receivers classical knowledge is known, which is not the case if the adversary has auxiliary input. Second, the *memory bound* is only enforced at *one* specific step in their protocol, while during the rest of the protocol, the adversary is allowed unlimited memory. The following very simple EPR-attack shows how the protocol can then be broken: We let the adversary receive an arbitrary number of halves of EPR-pairs from the environment as his auxiliary input, and run the protocol as before. Then, just before the memory bound is applied, he *teleports* his whole quantum memory to the environment. The classical communication needed to teleport can be part of the adversaries classical storage that he later outputs. Thus, the adversary can artificially increase his own storage by borrowing some quantum memory from the environment.

One possibility to overcome the second problem is to limit the memory of the environment. Yet, this solution seems very unsatisfactory: While we may be willing to accept that, say, a smart-card cannot store more than 100 qubits, this is much less clear for the environment. How could we place any limitations on the environment at all? In our framework, we thus always allow the environment to have an arbitrary amount of quantum memory, but limit the adversaries memory.

**Composable Security in the Bounded-Quantum-Storage Model.** We start by presenting a formal model for secure two-party computation in the bounded-quantum-storage model. Our model is quite similar to the model presented in [38], and provides *offline-security*. Then, we show that our model implies that secure protocols are sequentially composable.

Second, we slightly modify the model from [16] and prove the security of randomized OT in our refined model, which implies that the protocol is composable. In particular, we introduce a second memory bound into the protocol, which limits the amount of quantum auxiliary input the adversary may receive. We show that the simulator can extract the choice bit, even if the auxiliary quantum input remains completely unknown to him, and that the protocol is secure even if the quantum memory of the environment is unbounded. It turns out that the protocol only remains secure for a smaller memory bound in our model.

Third, we give well-known *classical* reductions of BC and OT to randomized OT in the appendix, and prove that they are secure in our model. Using the idea from [3], this also implies that the two players can *precompute* ROT, and, at a later point in time, they can use it to implement either an OT or a BC, for which they only need classical communication.

Since the proof presented in [40] carries over to our model, secure function evaluation in the bounded-quantum-storage model can be achieved by simply using the (classical) universal composable protocols presented in [18], which are based on [15]. Note that because our implementation of OT is physical, the results presented in [12] cannot be applied.

**Outline** In Section 2, we introduce the basic tools that we need later. In Section 3, we define a framework that provides offline security in the bounded-quantum-storage model, which implies that protocols can be composed sequentially. In Section 4, we then prove the security of the randomized oblivious transfer protocol from [16] in our refined model. In the appendix, we show that secure implementations of oblivious transfer and bit commitment follow by a (classical) reduction to randomized oblivious transfer.

## 2 Preliminaries

### 2.1 Notation

We assume general familiarity with the quantum model [22]. Throughout this paper, we use the term *computational basis* to refer to the basis given by  $\{|0\rangle, |1\rangle\}$ . We write  $+$  for the computational basis, and let  $|0\rangle_+ = |0\rangle$  and  $|1\rangle_+ = |1\rangle$ . The *Hadamard basis* is denoted by  $\times$ , and given by  $\{|0\rangle_\times, |1\rangle_\times\}$ , where  $|0\rangle_\times = (|0\rangle + |1\rangle)/\sqrt{2}$  and  $|1\rangle_\times = (|0\rangle - |1\rangle)/\sqrt{2}$ . For a string  $x \in \{0, 1\}^n$  encoded in bases  $b \in \{+, \times\}^n$ , we write  $|x\rangle_b = |x_1\rangle_{b_1}, \dots, |x_n\rangle_{b_n}$ . We also use 0 to denote  $+$ , and 1 to denote  $\times$ . Finally, we use  $x|_c$  to denote the sub-string of  $x$  consisting of all  $x_i$  where  $b_i = c$ .

We use the font  $\mathcal{A}$  to label a quantum register, corresponding to a Hilbert space  $\mathcal{A}$ . A *quantum channel* from  $\mathcal{A}$  to  $\mathcal{B}$  is a completely positive trace preserving (CPTP) map  $\Lambda : \mathcal{A} \rightarrow \mathcal{B}$ . We also call a map from  $\mathcal{A}$  to itself a *quantum operation*. Any quantum operation on the register  $\mathcal{A}$  can be phrased as a unitary operation on  $\mathcal{A}$  and an additional ancilla register  $\mathcal{A}'$ , where we trace out  $\mathcal{A}'$  to obtain the actions of the quantum operation on register  $\mathcal{A}$  [22]. We use  $\mathbb{S}(\mathcal{A})$  to refer to the set of all quantum states in  $\mathcal{A}$ , and  $\mathbb{T}(\mathcal{A})$  to refer to the set of all Hermitian matrices in  $\mathcal{A}$ . We use  $\mathbf{U}$  to refer to a quantum operation, upper case letters  $X$  to refer to classical random variables, the font  $\mathbb{S}$  for a set, and the font  $\mathbf{A}$  to refer to a player in the protocol.

### 2.2 Distance Measures

Our ability to distinguish two quantum states is determined by their *trace distance*. The trace distance between two states  $\rho, \rho' \in \mathbb{S}(\mathcal{H})$  is defined as  $D(\rho, \rho') := \frac{1}{2} \text{Tr} |\rho - \rho'|$ , where  $|A| = \sqrt{A^\dagger A}$ .

We also write  $\rho \equiv_\varepsilon \rho'$ , if  $D(\rho, \rho') \leq \varepsilon$ . For all practical purposes,  $\rho \equiv_\varepsilon \rho'$  means that the state  $\rho'$  behaves like the state  $\rho$ , except with probability  $\varepsilon$  [35]. For any quantum channel  $\Lambda$ , we have  $D(\Lambda(\rho), \Lambda(\rho')) \leq D(\rho, \rho')$ . Furthermore, the triangle inequality holds, i.e., for all  $\rho$ ,  $\rho'$  and  $\rho''$ , we have  $D(\rho, \rho'') \leq D(\rho, \rho') + D(\rho', \rho'')$ . Let  $\Lambda, \Lambda' : \mathcal{A} \rightarrow \mathcal{B}$  be two quantum channels. If for all  $\rho \in \mathcal{A}$ , we have  $\Lambda(\rho) \equiv_\varepsilon \Lambda'(\rho)$ , we may also write  $\Lambda \equiv_\varepsilon \Lambda'$ . Let  $\rho_{AB} \in \mathbb{S}(\mathcal{A} \otimes \mathcal{B})$  be classical on  $\mathcal{A}$ , i.e.  $\rho_{AB} = \sum_{x \in \mathcal{X}} P_X(x) |x\rangle\langle x| \otimes \rho_x$  for some distribution  $P_X$  over a finite set  $\mathcal{X}$ . We say that  $A$  is  $\varepsilon$ -close to uniform with respect to  $B$ , if  $D(\rho_{AB}, \mathbb{I}_A/d \otimes \rho_B) \leq \varepsilon$ , where  $d = \dim(\mathcal{H}_A)$ .

### 2.3 Uncertainty Relation and Privacy Amplification

For random variables  $X$  and  $Y$  with joint distribution  $P_{XY}$ , the *smooth conditional min-entropy* [36] can be expressed in terms of an optimization over events  $\mathcal{E}$  with probability at least  $1 - \varepsilon$ . Let  $P_{X\mathcal{E}|Y=y}(x)$  be the probability that  $\{X = x\}$  and  $\mathcal{E}$  occur conditioned on  $Y = y$ . We have

$$H_{\min}^\varepsilon(X|Y) = \max_{\mathcal{E}: \Pr(\mathcal{E}) \geq 1 - \varepsilon} \min_y \min_x (-\log P_{X\mathcal{E}|Y=y}(x)).$$

The smooth min-entropy allows us to use the following chain rule which does not hold in the case of standard min-entropy.

**Lemma 2.1** (Chain Rule [8, 28, 36]). *Let  $X$ ,  $Y$ , and  $Z$  be arbitrary random variables over  $\mathbb{X}$ ,  $\mathbb{Y}$  and  $\mathbb{Z}$ . Then for all  $\varepsilon, \varepsilon' > 0$ ,*

$$H_{\min}^{\varepsilon+\varepsilon'}(X|YZ) \geq H_{\min}^\varepsilon(XY|Z) - \log |\mathbb{Y}| - \log(1/\varepsilon').$$

We also need the following monotonicity of the smooth min-entropy

$$H_{\min}^\varepsilon(XY|Z) \geq H_{\min}^\varepsilon(X|Z).$$

A function  $h : \mathbb{S} \times \mathbb{X} \rightarrow \{0, 1\}^\ell$  is called a *two-universal hash function* [13], if for all  $x_0 \neq x_1 \in \mathbb{X}$ , we have  $\Pr[h(S, x_0) = h(S, x_1)] \leq 2^{-\ell}$  if  $S$  is uniform over  $\mathbb{S}$ . We thereby say that a random variable  $S$  is *uniform over* a set  $\mathbb{S}$  if  $S$  is chosen from  $\mathbb{S}$  according to the uniform distribution. For example, the class of all functions from  $\mathbb{S} \times \mathbb{X} \rightarrow \{0, 1\}^\ell$  is two-universal. *Privacy amplification* shows a two-universal hash function can be used to extract an almost random string from a source with enough min-entropy. The following theorem is from [16], stated slightly differently than the original statements in [35, 34].

**Theorem 2.2** (Privacy Amplification [35, 34]). *Let  $X$  and  $Z$  be (classical) random variables distributed over  $\mathbb{X}$  and  $\mathbb{Z}$ , and let  $Q$  be a random state of  $q$  qubits. Let  $h : \mathbb{S} \times \mathbb{X} \rightarrow \{0, 1\}^\ell$  be a two-universal hash function and let  $S$  be uniform over  $\mathbb{S}$ . If*

$$\ell \leq H_{\min}^{\varepsilon'}(X|Z) - q - 2\log(1/\varepsilon)$$

, then  $h(S, X)$  is  $(\varepsilon + 2\varepsilon')$ -close to uniform with respect to  $(S, Z, Q)$ .

The following lemma follows from the uncertainty relation presented in [16] by a simple purification argument and by fixing the parameter  $\lambda$  such that the error is at most  $\varepsilon$ . (see Appendix)

**Lemma 2.3.** Let  $X \in \{0, 1\}^n$  be a uniform random string, let  $B \in \{+, \times\}^n$  be a uniform random basis. Let  $|X\rangle_B = (|X_1\rangle_{B_1}, \dots, |X_n\rangle_{B_n})$  be a state of  $n$  qubits, and let  $K$  be the outcome of an arbitrary measurement of  $|X\rangle_B$ , which does not depend on  $X$  and  $B$ . Then, for any  $\varepsilon$ , we have

$$H_{\min}^\varepsilon(X|BK) \geq \frac{n}{2} - 10\sqrt[3]{n^2 \log \frac{1}{\varepsilon}},$$

which is positive if  $n > 8000 \log(1/\varepsilon)$ .

### 3 Security in the Bounded-Quantum-Storage Model

We now give a definition of offline-security in the bounded-quantum-storage model, and show that it allows protocols to be composed *sequentially*<sup>1</sup>. Our definitions are closely related to [38].

First of all, we assume that there is a global clock, that divides time into discrete rounds. We look at the following setting: Two *players*, A and B, execute a *protocol*  $\mathbf{P} = (\mathbf{P}_A, \mathbf{P}_B)$ , where  $\mathbf{P}_A$  is the program executed by A and  $\mathbf{P}_B$  the program executed by B. Before the first round, each program receives an input (that might be entangled with the input of the other player) and stores it. In each round, each program may first send/receive messages to/from a given functionality  $\mathbf{G}$ , then apply a quantum operation to its current internal storage (including the message space), and finally send/receive further messages at the end of each round.  $\mathbf{G}$  defines the communication resources available between the players, modeled as an interactive quantum functionality. It may contain a classical and/or a quantum communication channel, or other functionalities such as oblivious transfer or bit commitment. Finally, in the last step of the protocol each program outputs an output value. Note that the execution of  $\mathbf{P}$  using  $\mathbf{G}$  — denoted by  $\mathbf{P}(\mathbf{G})$  — is a quantum channel, which takes the input of both parties to the output of both parties. We also use the term *interface* of a player, to denote the interface presented by his program.

Players may be *honest*, which means that they follow the protocol, or they may be *corrupted*. All corrupted players belong to the *adversary*,  $A \subset \{A, B\}$ . We ignore the case where both players are corrupted, and we assume that this set is *static*, i.e., it is already fixed before the protocol starts. We only consider the case where the adversary is *active*, i.e., the adversary may not follow the protocol. The adversary  $A = \{p\}$  may replace his part of the protocol  $\mathbf{P}_p$  by another program  $\mathbf{A}_p$ . Opposed to  $\mathbf{P}_p$ ,  $\mathbf{A}_p$  receives some *auxiliary (quantum) input* in the first round that may also be entangled with the environment. We do not restrict the computational power of  $\mathbf{A}_p$  in any way, however we do limit its internal quantum storage to a certain *memory-bound* of  $m$  qubits. We call such a  $\mathbf{A}_p$  *m-bounded*.  $\mathbf{A}_p$  is allowed to perform arbitrary quantum operations in each round of the protocol. However after receiving his input, and after every round, all of his internal memory is measured, except for  $m$  qubits.<sup>2</sup>  $\mathbf{A}_p$  is not allowed to input or output any additional data *during* the execution of the protocol. The execution of  $\mathbf{P}$  using  $\mathbf{G}$ , where  $\mathbf{P}_p$  has been replaced by  $\mathbf{A}_p$ , is again a quantum channel, which maps the inputs of both players and the auxiliary input of the adversary to the outputs produced by both programs.

The *ideal functionality* defines what functionality we expect the protocol to implement. For the moment we only consider *non-interactive* functionalities, i.e., both players can send it input only

---

<sup>1</sup>Sequentially means that any given time only one sub-protocol is executed.

<sup>2</sup>Note that we enforce the memory bound after *every* round to keep the model simple. Later, in the security proof of our randomized OT protocol, we see that the bound needs only to be enforced twice. A practical implementation may introduce a wait time at these points to make sure the quantum memory physically decoheres.

once at the beginning, and obtain the output only once at the end. These functionalities have the form of a quantum channel. To make the definitions more flexible, we allow  $\mathbf{F}$  to look differently depending on whether both players are honest, or either A or B belongs to the adversary. So the ideal functionality is in fact a *collection of functionalities*,  $\mathbf{F} = (\mathbf{F}_\emptyset, \mathbf{F}_{\{A\}}, \mathbf{F}_{\{B\}})$ .  $\mathbf{F}_\emptyset$  denotes the functionality for the case when both players are honest, and  $\mathbf{F}_{\{A\}}$  and  $\mathbf{F}_{\{B\}}$  for the cases when A or B respectively are dishonest. We require that the honest player must always have the same interface as in  $\mathbf{F}_\emptyset$ , i.e., in  $\mathbf{F}_{\{A\}}$ , B must have the same interfaces as in  $\mathbf{F}_\emptyset$ , and in  $\mathbf{F}_{\{B\}}$ , A must have the same interfaces as in  $\mathbf{F}_\emptyset$ . We also require that  $\mathbf{F}_{\{A\}}$  and  $\mathbf{F}_{\{B\}}$  allow the adversary to play honestly, i.e., they must be at least as good for the adversary as the functionality  $\mathbf{F}_\emptyset$ .

We say that a protocol  $\mathbf{P}$  having access to the functionality  $\mathbf{G}$ <sup>3</sup> securely implements a functionality  $\mathbf{F}$ , if the following conditions are satisfied: First of all, we require that the protocol has almost the same output as  $\mathbf{F}$ , if both players are honest. Second, for  $\mathbb{A} = \{p\}$ , we require that the adversary attacking the protocol has basically no advantage over attacking  $\mathbf{F}$  directly. We thus require that for every  $m$ -bounded program  $\mathbf{A}_p$ , there exists a  $s$ -bounded program  $\mathbf{S}_p$  (called the *simulator*), such that the overall outputs of both situations is almost the same, for all inputs. For simplicity, we do not make any restrictions on the efficiency of the simulators<sup>4</sup>. Also, we do not require him to use the adversary  $\mathbf{A}_p$  as a black-box:  $\mathbf{S}_p$  may be constructed from scratch, under full knowledge of the behaviour of  $\mathbf{A}_p$ . In particular, we allow him to execute some or all actions of  $\mathbf{A}_p$  in a single round. Recall, that a memory bound is applied only after each round. Thus, when executing  $\mathbf{A}_p$  in a single round, the simulator will not experience any memory bound. This model is motivated by the physically realistic assumption that such memory bounds are introduced by adding specific waiting times after each round. Hence, this does not give the simulator any memory. However, in order to make protocols composable with other protocol in our model, we do require the simulator to be memory-bounded as well. The amount of memory required by the simulator gives a bound on the *virtual* memory the adversary seems to have by attacking the real protocol instead of the ideal one. Ideally, we would like  $\mathbf{S}_p$  to use the same amount of memory as  $\mathbf{A}_p$ . The simulator  $\mathbf{S}_p$  can be represented by two quantum channels. The first channel maps the input and the auxiliary input to an input to the ideal functionality, and to a state of at most  $s$  qubit. The other channel maps that state and the output of the ideal functionality to the output of the simulator.

**Definition 3.1.** *A protocol  $\mathbf{P}(\mathbf{F}) = (\mathbf{P}_A, \mathbf{P}_B)(\mathbf{F})$  implements  $\mathbf{G}$  with an error of at most  $\varepsilon$ , secure against  $m$ -bounded adversaries using  $s$ -bounded simulators, if*

- (Correctness)  $\mathbf{P}(\mathbf{F}_\emptyset) \equiv_\varepsilon \mathbf{G}_\emptyset$  .
- (Security for A) For every  $m$ -bounded  $\mathbf{A}_B$  there exists a  $s$ -bounded  $\mathbf{S}_B$ , such that

$$(\mathbf{P}_A, \mathbf{A}_B)(\mathbf{F}_{\{B\}}) \equiv_\varepsilon \mathbf{S}_B(\mathbf{G}_{\{B\}}) .$$

- (Security for B) For every  $m$ -bounded  $\mathbf{A}_A$  there exists a  $s$ -bounded  $\mathbf{S}_A$ , such that

$$(\mathbf{A}_A, \mathbf{P}_B)(\mathbf{F}_{\{A\}}) \equiv_\varepsilon \mathbf{S}_A(\mathbf{G}_{\{A\}}) .$$

---

<sup>3</sup> $\mathbf{G}$  may also be a collection of functionalities.

<sup>4</sup>Recall that the adversary is computationally unbounded as well.

An important property of our definition is that it allows protocols to be composed. The following theorem shows that in a secure protocol that is based on an ideal, non-interactive functionality  $\mathbf{G}$  and some other functionalities  $\mathbf{G}'$ <sup>5</sup>, we can replace  $\mathbf{G}$  with a secure implementation of  $\mathbf{G}$ , without making the protocol insecure. The theorem requires that  $\mathbf{G}$  is called sequentially, i.e., that no other subprotocols are running parallel to  $\mathbf{G}$ . The proof uses the same idea as in the classical case [10].

**Theorem 3.2** (Sequential Composition Theorem). *Let  $\mathbf{F}$  and  $\mathbf{G}$  be non-interactive functionalities, and  $\mathbf{G}'$  and  $\mathbf{H}$  be arbitrary functionalities. Let  $\mathbf{P}(\mathbf{G}\|\mathbf{G}')$  be a protocol that calls  $\mathbf{G}$  sequentially and that implements  $\mathbf{F}$  with error of at most  $\varepsilon_1$  secure against  $m_1$ -bounded adversaries using  $s_1$ -bounded simulators, and let  $\mathbf{Q}(\mathbf{H})$  be a protocol that implements  $\mathbf{G}$  with error of at most  $\varepsilon_2$  secure against  $m_2$ -bounded adversaries using  $s_2$ -bounded simulators, where  $m_2 \geq s_1$ . Then  $\mathbf{P}(\mathbf{Q}(\mathbf{H})\|\mathbf{G}')$  implements  $\mathbf{F}$  with error at most  $\varepsilon_1 + \varepsilon_2$ , secure against  $\min(m_1, m_2)$ -bounded adversaries using  $s_2$ -bounded simulators.*

*Proof.* (Sketch) If both players are honest, the statement follows directly from the properties of the trace distance, since we have  $\mathbf{P}(\mathbf{G}_\emptyset\|\mathbf{G}'_\emptyset) \equiv_{\varepsilon_1} \mathbf{P}(\mathbf{Q}(\mathbf{H}_\emptyset)\|\mathbf{G}'_\emptyset)$ , and hence  $\mathbf{F}_\emptyset \equiv_{\varepsilon_1 + \varepsilon_2} \mathbf{P}(\mathbf{Q}(\mathbf{H}_\emptyset)\|\mathbf{G}'_\emptyset)$ .

Let  $\mathbf{A}$  be honest, and let  $\mathbf{B}$  attack the protocol  $\mathbf{P}(\mathbf{Q}(\mathbf{H})\|\mathbf{G}')$  by executing  $\mathbf{A}_\mathbf{B}$ . We cut  $\mathbf{A}_\mathbf{B}$  into three parts. Let  $\mathbf{A}_\mathbf{B}^{(0)}$  be executed before protocol  $\mathbf{Q}$  starts,  $\mathbf{A}_\mathbf{B}^{(1)}$  during  $\mathbf{Q}$ , and  $\mathbf{A}_\mathbf{B}^{(2)}$  after  $\mathbf{Q}$ . Since  $\mathbf{A}_\mathbf{B}^{(1)}$  is  $\min(m_1, m_2)$ -bounded and  $\mathbf{Q}$  is secure, there exist a  $s_1$ -bounded  $\mathbf{S}_\mathbf{B}^{(1)}$ , such that  $(\mathbf{Q}_\mathbf{A}, \mathbf{A}_\mathbf{B}^{(1)})(\mathbf{H}_{\{\mathbf{B}\}}) \equiv_{\varepsilon_2} \mathbf{S}_\mathbf{B}^{(1)}(\mathbf{G}_{\{\mathbf{B}\}})$ . Let  $\mathbf{A}'_\mathbf{B}$  be the program that results from joining  $\mathbf{A}_\mathbf{B}^{(0)}, \mathbf{S}_\mathbf{B}^{(1)}$ , and  $\mathbf{A}_\mathbf{B}^{(2)}$ . Because of  $\max(\min(m_1, m_2), s_1) \leq m_2$ ,  $\mathbf{A}'_\mathbf{B}$  is  $m_2$ -bounded and  $\mathbf{P}$  is secure, there exists a  $s_2$ -bounded  $\mathbf{S}_\mathbf{B}$ , such that  $(\mathbf{P}_\mathbf{A}, \mathbf{A}'_\mathbf{B})(\mathbf{G}_{\{\mathbf{B}\}}\|\mathbf{G}'_{\{\mathbf{B}\}}) \equiv_{\varepsilon_1} \mathbf{S}_\mathbf{B}(\mathbf{F}_{\{\mathbf{B}\}})$ . It follows now from the properties of the trace distance that  $\mathbf{S}_\mathbf{B}$  is a simulator that satisfies the security condition for  $\mathbf{A}$  with an error of at most  $\varepsilon_1 + \varepsilon_2$ . The security for  $\mathbf{B}$  can be shown in the same way.  $\square$

**Interactive functionalities.** The definitions above only apply to non-interactive functionalities, i.e. functionalities that consist of just one input/output phase. In general, we would also like to securely implement functionalities with several such phases. The most prominent example of such a functionality is *bit-commitment*, which has two phases, a *commit-phase*, and an *open-phase*.

The security definitions and the composition theorem generalize to the multi-phase case. Basically, all phases by themselves can be treated as individual, non-interactive functionalities, using the security definition given above. We can assume that the adversary always sends his internal classical and quantum state to the environment at the end of each phase, and receives it back at the beginning of the next phase. The adversary can thus be modeled by individual adversaries for each phase. However, since the ideal functionalities between the different phases are connected by some shared memory, i.e., the actions of the functionality in the second phase may depend on the actions in the first phase, the simulator must be allowed to use some *classical* memory *between* the rounds.

## 4 Randomized Oblivious Transfer

We now apply our framework to the randomized OT protocol presented in [17]. In particular, we prove security with respect to the following definition of randomized oblivious transfer. We show

---

<sup>5</sup>We denote the concatenation of the functionalities  $\mathbf{G}$  and  $\mathbf{G}'$  by  $\mathbf{G}\|\mathbf{G}'$ .

in the appendix how to obtain the standard notion of OT from randomized OT. Note that in our version of randomized OT, also the choice bit  $c$  of the receiver is randomized.

**Definition 4.1** (Randomized oblivious transfer).  $(\frac{2}{1})\text{-ROT}^\ell$  (or, if  $\ell$  is clear from the context,  $\text{ROT}$ ) is defined as  $\text{ROT} = (\text{ROT}_\emptyset, \text{ROT}_{\{A\}}, \text{ROT}_{\{B\}})$ , where

- $\text{ROT}_\emptyset$ : The functionality chooses uniformly at random the value  $(x_0, x_1) \in_R \{0, 1\}^\ell \times \{0, 1\}^\ell$  and  $c \in_R \{0, 1\}$ . It sends  $(x_0, x_1)$  to A and  $(c, y)$  to B where  $y = x_c$ .
- $\text{ROT}_{\{A\}}$ : The functionality receives  $(x_0, x_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  from A. Then, it chooses  $c \in_R \{0, 1\}$  uniformly at random and sends  $(c, y)$  to B, where  $y = x_c$ .
- $\text{ROT}_{\{B\}}$ : The functionality receives  $(c, y) \in \{0, 1\} \times \{0, 1\}^\ell$  from B. Then, it sets  $x_c = y$ , chooses  $x_{1-c} \in_R \{0, 1\}^\ell$  uniformly at random, and sends  $(x_0, x_1)$  to A.

We first briefly recall the protocol. The protocol  $\text{BQS-OT} = (\text{BQS-OT}_A, \text{BQS-OT}_B)$  uses a noiseless unidirectional quantum channel  $\text{Q-Comm}$ , and a noiseless unidirectional classical channel  $\text{Comm}$ , both from the sender to the receiver. Let  $h : \mathcal{R} \times \{0, 1\}^n \rightarrow \{0, 1\}^\ell$  be a two-universal hash function. The sender (A) and receiver (B) execute the following:

**Protocol 1: BQS-OT<sub>A</sub>**

1. Choose  $x \in_R \{0, 1\}^n$  and  $b \in_R \{0, 1\}^n$  uniformly at random.
2. Send  $|x\rangle_b := (|x_1\rangle_{b_1}, \dots, |x_n\rangle_{b_n})$  to  $\text{Q-Comm}$ , where  $|x_i\rangle_{b_i}$  is  $x_i$  encoded in the basis  $b_i$ .
3. Choose  $r_0, r_1 \in_R \mathcal{R}$  uniformly at random and send  $(b, r_0, r_1)$  to  $\text{Comm}$ .
5. Output  $(s_0, s_1) := (h(r_0, x|_0), h(r_1, x|_1))$ , where  $x|_j$  is the string of all  $x_i$  where  $b_i = j$ .

**Protocol 2: BQS-OT<sub>B</sub>**

1. Choose  $c \in_R \{0, 1\}$  uniformly at random.
2. Receive the qubits  $(q_1, \dots, q_n)$  from  $\text{Q-Comm}$  and measure them in the basis  $c$ , which gives output  $x' \in \{0, 1\}^n$ .
3. Receive  $(b, r_0, r_1)$  from  $\text{Comm}$ .
4. Output  $(c, y) := (c, h(r_c, x'|_c))$ , where  $x'|_c$  is the string of all  $x'_i$  where  $b_i = c$ .

Note that the values  $x|_0$ ,  $x|_1$  and  $x'|_c$  are in fact padded by additional 0 to have a length of  $n$  bits. This padding does not affect their entropies. A memory bound is applied before step 1, and before step 3 of the receiver.

**Security against the sender.** We first consider the case when the sender, A, is dishonest. This case turns out to be quite straightforward. In general, we can describe any action of the adversary by a unitary followed by a measurement in the computational basis. We use the following letters to

refer to the different classical and quantum registers available to the adversary: Let  $\mathcal{Q}$  denote the quantum register. Note that since we assume that our adversary's memory is  $m$ -bounded, the size of  $\mathcal{Q}$  does not exceed  $m$ . Let  $\mathcal{M}_Q$  and  $\mathcal{M}_K$  denote the quantum and classical registers, that hold the messages sent to the receiver. Let  $\mathcal{K}$  denote the classical input register of the adversary. Finally, let  $\mathcal{A}$  denote an auxiliary quantum register. Recall from Section 2, that any quantum operation on  $\mathcal{Q}$  and  $\mathcal{M}_Q$  can be implemented by a unitary followed by a measurement on an additional register  $\mathcal{A}$ . Wlog we let  $\mathcal{A}$  and  $\mathcal{M}_Q$  be measured in the computational basis to enforce a memory bound.

To model quantum and classical input that a malicious  $\mathbf{A}$  may receive, we let  $\mathcal{Q}$  start out in any state  $\rho_{\text{in}}$ , unknown to the simulator. Likewise,  $\mathcal{K}$  may contain some classical input  $k_{\text{in}}$  of  $\mathbf{A}$ . Wlog we assume that all other registers start out in a fixed state of  $|0\rangle$ . We can then describe the actions of  $\mathbf{A}$  by a single unitary  $\mathbf{A}_{\mathbf{A}}$  defined by

$$\mathbf{A}_{\mathbf{A}}(\underbrace{\rho_{\text{in}}}_{\mathcal{Q}} \otimes \underbrace{|0\rangle\langle 0|}_{\mathcal{A}} \otimes \underbrace{k_{\text{in}}}_{\mathcal{K}} \otimes \underbrace{|0\rangle\langle 0|}_{\mathcal{M}_Q} \otimes \underbrace{|0\rangle\langle 0|}_{\mathcal{M}_K}) \mathbf{A}_{\mathbf{A}}^\dagger = \underbrace{\rho_{\text{out}}}_{\mathcal{Q}, \mathcal{A}} \otimes \underbrace{k_{\text{in}}}_{\mathcal{K}} \otimes \underbrace{\rho_{x_b}}_{\mathcal{M}_Q} \otimes \underbrace{|b, r_0, r_1\rangle\langle b, r_0, r_1|}_{\mathcal{M}_K}. \quad (1)$$

Note that without loss of generality  $\mathbf{A}_{\mathbf{A}}$  leaves  $\mathcal{K}$  unmodified: since  $\mathcal{K}$  is classical we can always copy its contents to  $\mathcal{A}$  and let all classical output be part of  $\mathcal{A}$ . To enforce the memory bound, assume wlog that  $\mathcal{A}$  and  $\mathcal{M}_Q$  are now measured completely in the computational basis. We now show that for any adversary  $\mathbf{A}_{\mathbf{A}}$  there exists an appropriate simulator  $\mathbf{S}_{\mathbf{A}}$ .

**Lemma 4.2.** *Protocol BQS-OT is secure against dishonest  $\mathbf{A}$ .*

*Proof.* Let  $\mathbf{S}_{\mathbf{A}}$  be defined as follows:  $\mathbf{S}_{\mathbf{A}}$  runs  $\mathbf{A}_{\mathbf{A}}$ <sup>6</sup>, and measures register  $\mathcal{M}_Q$  in the basis determined by  $\mathcal{M}_K$ . This allows him to compute  $s_0 = h(r_0, x_{|0})$  and  $s_1 = h(r_1, x_{|1})$ .  $\mathbf{S}_{\mathbf{A}}$  then sends  $s_0$  and  $s_1$  to  $\text{ROT}_{\{\mathbf{A}\}}$ . It is clear that since the simulator based his measurement on  $\mathcal{M}_K$ ,  $s_0$  and  $s_1$  are consistent with the run of the protocol. Furthermore, note that  $\mathbf{S}_{\mathbf{A}}$  did not need to touch register  $\mathcal{Q}$  at all. We can thus immediately conclude that the environment can tell no difference between the real protocol and the ideal setting.  $\square$

**Security against the receiver.** To prove security against a dishonest receiver requires a more careful treatment of the quantum input given to the adversary. The main idea behind our proof is that the memory bound in fact *fixes* a classical bit  $c$ . Our main challenge is to find a  $c$  that the simulator can calculate and that is consistent with the adversary and his input, while keeping the output state of the adversary intact. To do so, we use a generalization of the *min-entropy splitting lemma* in [16], which in turn is based on an earlier version of [45]. It states that if two random variables  $X_0$  and  $X_1$  together have high min-entropy, than we can define a random variable  $C$ , such that  $X_{1-C}$  has least half of the original min-entropy. To find  $C$ , one must know the distributions of  $X_0$  and  $X_1$ . In the following generalization, we do *not* exactly know the distribution of  $X_0$  and  $X_1$ , since we assume that its distribution also depends on an unknown random variable  $J$ , distributed over a domain of the size  $2^\beta$ . Note that  $\beta = 0$  give the min-entropy splitting lemma in [16].

**Lemma 4.3** (Generalized Min-Entropy Splitting Lemma). *Let  $\varepsilon \geq 0$ , and  $0 < \beta < \alpha$ . Let  $J$  be a random variable over  $\{0, \dots, 2^\beta - 1\}$ , and let  $X_0$ ,  $X_1$  and  $K$  be random variables such that*

---

<sup>6</sup>As described in Section 3,  $\mathbf{S}_{\mathbf{A}}$  can effectively skip the wait time required for the memory bound to take effect, since he can execute  $\mathbf{A}_{\mathbf{A}}$  before his memory bound is applied.

$H_{\min}^\varepsilon(X_0X_1 \mid KJ) \geq \alpha$ . Let  $f(x_0, x_1, k) = 1$ , if there exists an  $j \in \{0, \dots, 2^\beta - 1\}$  such that  $P_{X_1|KJ}(x_1, k, j) \geq 2^{-(\alpha-\beta)/2}$ , and 0 otherwise, and let  $C := f(X_0, X_1, K)$ . We have

$$H_{\min}^\varepsilon(X_{1-C}C \mid KJ) \geq \frac{\alpha - \beta}{2}.$$

*Proof.* Let  $S_k^j$  be the set of values  $x_1$  for which  $P_{X_1|KJ}(x_1, k, j) \geq 2^{-(\alpha-\beta)/2}$ . We have  $|S_k^j| \leq 2^{(\alpha-\beta)/2}$ , since all values in  $S_k^j$  have a probability that is at least  $2^{-(\alpha-\beta)/2}$ . Let  $S_k := \bigcup_j S_k^j$ . We have  $|S_k| \leq 2^\beta \cdot 2^{(\alpha-\beta)/2} = 2^{(\alpha+\beta)/2}$ .

Let  $K = k$  and  $J = j$ . Because  $C = 0$  implies that  $X_1 \notin S_k$ , and thus also that  $X_1 \notin S_k^j$ , we have  $P_{X_1C|KJ}(x_1, 0, k, j) < 2^{(\alpha-\beta)/2}$ . It follows from the assumption that there exists an event  $\mathcal{E}$  with probability  $1 - \varepsilon$  such that for all  $x_0, x_1, k$  and  $j$ , we have  $P_{X_0X_1\mathcal{E}|KJ}(x_0, x_1, k, j) \leq 2^{-\alpha}$ . It follows that

$$P_{X_0C\mathcal{E}|KJ}(x_0, 1, k, j) = \sum_{x_1 \in S_k} P_{X_0X_1\mathcal{E}|KJ}(x_0, x_1, k, j) \leq 2^{(\alpha+\beta)/2} \cdot 2^{-\alpha} = 2^{-(\alpha-\beta)/2}.$$

The statement follows.  $\square$

We now describe the actions of the adversary. Let  $\mathcal{Q}$  denote his quantum storage register, and let  $\mathcal{A}$  denote an auxiliary quantum register as above. Again, the size of  $\mathcal{Q}$  does not exceed  $m$ . Let  $\mathcal{K}$  denote his classical input register, and let  $\mathcal{M}$  denote the register holding the quantum message he receives from the sender in step 2. Let  $\mathcal{E}$  denote the message register holding the classical messages he receives in step 3. Again, we assume that  $\mathcal{Q}$  is initialized to his quantum input state  $\rho_{\text{in}}$ . Likewise,  $\mathcal{K}$  is initialized to his classical input  $k_{\text{in}}$ . All other registers are initialized to  $|0\rangle$ . We can now describe the actions of the adversary by two unitaries, where a memory bound is applied after the first. The action of the adversary following step 2 can be described as a unitary  $\mathbf{A}_B^{(1)}$  similar to Eq. 1. Note we can again assume that  $\mathbf{A}_B^{(1)}$  leaves  $\mathcal{K}$  unmodified. To enforce the memory bound, we now let register  $\mathcal{M}$  and  $\mathcal{A}$  be measured in the computational basis. We use  $\rho_{\text{out}} \in \mathcal{Q}$  to denote the adversary's quantum output, and  $k_{\text{out}} \in \mathcal{M} \otimes \mathcal{A}$  to denote his classical output. After the memory bound is applied, the receiver obtains additional information from the sender. The actions of the adversary after step 3 can then be described by a unitary  $\mathbf{A}_B^{(2)}$  followed by a measurement of quantum registers  $\mathcal{M}$  and  $\mathcal{A}$  in the computational basis.

In order to make the proof easier to understand, we build it up in 3 steps: First, we analyze the easy case where there is no quantum auxiliary input, which is essentially equivalent to the original security proof. Then we extend it, by allowing the adversary some quantum auxiliary input of size  $\beta$ , pure and mixed. We start with  $\beta = 0$ , but keep  $\beta$  as a parameter, so that we can later generalize the statement.

**Lemma 4.4.** *Protocol BQS-OT is secure against dishonest  $\mathcal{B}$  with an error of at most  $5\varepsilon$ , if he receives no quantum (auxiliary) input, and his quantum memory is bounded before step 1 and between step 2 and 3 by  $m$  qubits, for*

$$8\ell + 2\beta + 4m \leq n - 20\sqrt[3]{n^2 \log \frac{1}{\varepsilon}} - 12 \log \frac{1}{\varepsilon} - 4,$$

where  $\beta$  is a parameter.

*Proof.* Let  $K_{\text{in}}$  be the classical auxiliary input the adversary receives, and let  $|\Psi_{\text{in}}\rangle = |j\rangle$  be the auxiliary quantum input for some fixed  $j$  known to the simulator. First of all, the simulator simulates the actions of the sender following steps 1 and 2, using a random string  $X$  and a random basis  $B$ . The simulator then applies  $\mathbf{A}_B^{(1)}$ , which gives him some classical output  $K_{\text{out}}$ , and a quantum state  $\rho_{\text{out}}$ . It follows from the uncertainty relation of Lemma 2.3 that

$$H_{\min}^{2\varepsilon}(X \mid BK_{\text{out}}K_{\text{in}}) \geq \alpha,$$

for  $\alpha := n/2 - 10\sqrt[3]{n^2 \log(1/\varepsilon)}$ . Let  $(X_0, X_1) := X$ , where  $X_0 := X_{|0}$  and  $X_1 := X_{|1}$  are the substrings of  $X$  defined in the same way as in the protocol. Note that since the simulator holds a description of  $|\Psi_{\text{in}}\rangle$  and  $\mathbf{A}^{(1)}$ , he knows  $P_{X_0 X_1 B K_{\text{out}} K_{\text{in}}}$ , and thus we can apply Lemma 4.3, for  $K = (B, K_{\text{out}}, K_{\text{in}})$  and a constant  $J$  (or,  $\beta = 0$ ). Since the simulator knows the values  $X_0, X_1$  and  $K$ , he can calculate the value  $C := f(X_0, X_1, K)$ , for which we have

$$H_{\min}^\varepsilon(X_{1-C}C \mid K) \geq \frac{\alpha - \beta}{2}.$$

The simulator now chooses  $R_0$  and  $R_1$  uniformly at random and calculates  $S_0 = h(R_0, X_0)$  and  $S_1 = h(R_1, X_1)$ . Since  $R_0$  and  $R_1$  are independent of  $X_0, X_1$  and  $C$ , we have

$$H_{\min}^\varepsilon(X_{1-C}C \mid K) = H_{\min}^\varepsilon(X_{1-C}C \mid R_C K).$$

Using the chain rule from Lemma 2.1 and the monotonicity of  $H_{\min}^\varepsilon$ , we obtain

$$\begin{aligned} H_{\min}^{2\varepsilon}(X_{1-C} \mid CR_C K S_C) &\geq H_{\min}^\varepsilon(X_{1-C} S_C C \mid R_C K) - (\ell + 1) - \log \frac{1}{\varepsilon} \\ &\geq H_{\min}^\varepsilon(X_{1-C} C \mid R_C K) - (\ell + 1) - \log \frac{1}{\varepsilon} \\ &\geq \frac{\alpha - \beta}{2} - \ell - 1 - \log \frac{1}{\varepsilon}. \end{aligned}$$

By using the privacy amplification Theorem 2.2, we get that  $S_{1-C}$  is  $5\varepsilon$  close to uniform with respect to  $(R_0, R_1, C, S_C, B, K_{\text{out}}, K_{\text{in}})$  and  $\rho_{\text{out}}$  if

$$\ell \leq \frac{\alpha - \beta}{2} - \ell - 1 - \log \frac{1}{\varepsilon} - m - 2 \log \frac{1}{\varepsilon}.$$

By replacing  $\alpha$  and rearranging the terms we get the claimed equation.

The simulator now sets  $Y := S_C$ , and sends  $(C, Y)$  to  $\text{ROT}_{\{B\}}$ . To complete the simulation, he runs  $\mathbf{A}_A^{(2)}$  as the adversary would have. Note that the simulator did not require any more memory than the adversary itself, i.e., we can take  $\mathbf{S}_B$  to be  $m$ -bounded as well. Clearly, the simulator determined  $C$  solely from the classical output of the adversary and thus the adversary's output state in the simulated run is equal to the original output state of the adversary  $\rho_{\text{out}} \otimes k_{\text{out}}$ . Since the only difference between the simulation and the real execution is that in the simulation,  $S_{1-C}$  is chosen completely at random, the simulation is  $5\varepsilon$ -close to the output of the real protocol.  $\square$

We now show how to extend the above analysis to the case where the adversary's input is pure. Note that if the adversary's input is pure, the adversary cannot be entangled with the environment.

**Lemma 4.5.** *Protocol BQS-OT is secure against dishonest B with an error of at most  $5\varepsilon$ , if he receives a pure state quantum (auxiliary) input, and his quantum memory is bounded before step 1 by  $\beta$  qubits, and between step 2 and 3 by  $m$  qubits, for*

$$8\ell + 2\beta + 4m \leq n - 20\sqrt[3]{n^2 \log \frac{1}{\varepsilon}} - 12 \log \frac{1}{\varepsilon} - 4.$$

*Proof.* Let  $|j\rangle$  for  $j \in \{0, \dots, 2^\beta\}$  be a basis for the quantum auxiliary input. Any fixed auxiliary input  $|j\rangle$  and  $k_{\text{in}}$  fixes a distribution  $P_{X_0 X_1 K | J=j}$ , where  $K$  is the classical value the adversary has after second memory bound. Using the same argumentation as in Lemma 4.4, but now using the generalized min-entropy splitting lemma with  $\beta > 0$ , we can construct a simulator that does not need to know  $j$ , nor the distribution  $P_J$ .

Hence, the simulator can construct a linear transformation acting on registers  $\mathcal{Q}, \mathcal{M}, \mathcal{A}, \mathcal{K}, \mathcal{X}, \mathcal{B}, \mathcal{R}$ , and  $\mathcal{C}$  combining the actions of  $\mathbf{A}_A^{(1)}$  and the extraction of  $c$  using the function  $f$  as defined above. We have

$$\begin{aligned} \mathbf{S}_B^2 \left( \sum_j \alpha_j \underbrace{|j\rangle}_{\mathcal{Q}} \otimes \underbrace{|x_b\rangle}_{\mathcal{M}} \otimes \underbrace{|0\rangle}_{\mathcal{A}} \otimes \underbrace{|k_{\text{in}}\rangle}_{\mathcal{K}} \otimes \underbrace{|x\rangle}_{\mathcal{X}} \otimes \underbrace{|b\rangle}_{\mathcal{B}} \otimes \underbrace{|r_0, r_1\rangle}_{\mathcal{R}} \otimes \underbrace{|0\rangle}_{\mathcal{C}} \otimes \underbrace{|0\rangle}_{\mathcal{Y}} \right) = \\ \sum_{q, m_1, a_1} \alpha_{q, m_1, a_1} \underbrace{|q\rangle}_{\mathcal{Q}} \otimes \underbrace{|m_1\rangle}_{\mathcal{M}} \otimes \underbrace{|a_1\rangle}_{\mathcal{A}} \otimes \underbrace{|k_{\text{in}}\rangle}_{\mathcal{K}} \otimes \underbrace{|x\rangle}_{\mathcal{X}} \otimes \underbrace{|b\rangle}_{\mathcal{B}} \otimes \underbrace{|r_0, r_1\rangle}_{\mathcal{R}} \otimes \underbrace{|c\rangle}_{\mathcal{C}} \otimes \underbrace{|s_0, s_1\rangle}_{\mathcal{Y}} \end{aligned}$$

for any pure state input  $|\Psi_{\text{in}}\rangle = \sum_j \alpha_j |j\rangle$ . Wlog, all registers except  $\mathcal{Q}$  are now measured in the computational basis as the memory bound takes effect. The input state  $|\Psi_{\text{in}}\rangle$  will define the distribution of  $J$  for the generalized min-entropy splitting lemma. The rest follows as above.  $\square$

It remains to address the case where the receiver gets a mixed state quantum input. This is the case where the adversary receives a state that is entangled with the environment. Note that this means that we must decrease the size of the adversary's memory: If he could receive an entangled state of  $\beta$  qubits as input, he could use it to increase his memory to  $m + \beta$  qubits by teleporting  $\beta$  qubits to the environment, and storing the remaining  $m$ . Hence, we now have to take the adversary to be  $m'$ -bounded, where  $m' := m - \beta$ . Luckily, using a similar argument as in [43], we can now extend the argument given above: Note that for any pure state input  $|\Psi\rangle = |\Psi_{\text{in}}\rangle \otimes k_{\text{in}}$ , the output of the simulated adversary is *exactly*  $\Lambda(|\Psi\rangle\langle\Psi|)$ , where  $\Lambda$  is the adversary's channel. Since  $\{|\Psi\rangle\langle\Psi| \mid |\Psi\rangle \in \mathcal{Q} \otimes \mathcal{K}, \|\Psi\| = 1\}$  spans all of  $\mathbb{T}(\mathcal{Q} \otimes \mathcal{K})$  and the map given by the simulation procedure is the same as  $\Lambda$  on all inputs, we can conclude that the complete map is equal to  $\Lambda$ . Note that the simulator does not need to consider the  $\beta$  qubits that the adversary might have teleported to the environment: we can essentially view it as part of the original adversary's quantum memory, and the simulator bases his decision solely on the classical output of the adversary. Hence:

**Lemma 4.6.** *Protocol BQS-OT is secure against dishonest B with an error of at most  $5\varepsilon$ , if he receives a quantum (auxiliary) input, and his quantum memory is bounded before step 1 by  $\beta$  qubits and between step 2 and 3, by  $m$  qubits, for*

$$8\ell + 6\beta + 4m \leq n - 20\sqrt[3]{n^2 \log \frac{1}{\varepsilon}} - 12 \log \frac{1}{\varepsilon} - 4.$$

The following theorem follows now directly from Lemma 4.2 and 4.6.

**Theorem 4.7.** *Protocol  $BQS\text{-}OT(Q\text{-}\mathit{Comm} \parallel \mathit{Comm})$  implements  $\binom{2}{1}\text{-}\mathit{ROT}^\ell$  with an error of at most  $5\varepsilon$ , secure against  $m$ -bounded adversaries using  $m$ -bounded simulators, if*

$$8\ell + 10m \leq n - 20\sqrt[3]{n^2 \log \frac{1}{\varepsilon}} - 12 \log \frac{1}{\varepsilon} - 4.$$

Note that there are ways to improve on these parameters. For example, the splitting lemma defines the function  $f$  in an asymmetric way, which implies that for  $C = 1$ , in fact the bound also holds for the *conditional* min-entropy of  $X_0$  given  $X_1$ . Thus, we would not need to additionally apply the chain rule for this case. We did not do this here to keep the proof simple.

#### 4.1 On Parallel Composition

For efficiency, it would be important to know if the protocol from last section would also be secure under *parallel* composition. Unfortunately, this is not an easy task: First, consider executing the protocol in parallel, when the sender and receiver are the same for each instance of the protocol. Clearly, the overall memory of the committer cannot exceed the amount of memory he would be allowed for a single execution of the protocol: otherwise he could cheat in at least one instance of the protocol. However, even when imposing such a constraint, parallel composition remains tricky: Second, consider the case where we run two instances of the protocol in parallel, where the roles of the sender (initially Alice) and the receiver (initially Bob) are *exchanged* in the second instance of the protocol. Let the malicious Bob behave as follows: Upon reception of the quantum states in the first instance of the protocol, he immediately returns them unmeasured to Alice. Later, he sends the very same values  $(b, r_0, r_1)$  back to Alice. Alice thus measures her own states, in her own bases. Thus, her output  $y$  of the second instance of the protocol will always be equal either to  $x_0$  or to  $x_1$  of the first instance of the protocol. This is clearly something Bob would not be able to do in an ideal setting. This simple example already shows that great care must be taken when composing such protocols in parallel: no quantum memory was required to execute such an attack.

### 5 Acknowledgments

We thank Simon Pierre Desrosiers and Christian Schaffner for useful comments, and Dominique Unruh for a kind explanation of his work.

### References

- [1] M. Backes, B. Pfitzmann, and M. Waidner. A universally composable cryptographic library. <http://eprint.iacr.org/2003/015>, 2003.
- [2] D. Beaver. Foundations of secure interactive computing. In *Advances in Cryptology — CRYPTO '91*, volume 1233 of *Lecture Notes in Computer Science*, pages 377–391. Springer-Verlag, 1992.
- [3] D. Beaver. Precomputing oblivious transfer. In *Advances in Cryptology — EUROCRYPT '95*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109. Springer-Verlag, 1995.

- [4] M. Ben-Or and D. Mayers. General security definition and composability for quantum and classical protocols. quant-ph/0409062, 2004.
- [5] C. H. Bennett, G. Brassard, C. Crépeau, and H. Skubiszewska. Practical quantum oblivious transfer. In *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 351–366. Springer, 1992.
- [6] M. Blum. Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, 15(1):23–27, 1983.
- [7] H. Buhrman, M. Christandl, P. Hayden, H.-K. Lo, and S. Wehner. Security of quantum bit string commitment depends on the information measure. *Physical Review Letters*, 97:250501, 2006.
- [8] C. Cachin. *Entropy Measures and Unconditional Security in Cryptography*. PhD thesis, ETH Zurich, Switzerland, 1997.
- [9] C. Cachin, C. Crépeau, and J. Marcil. Oblivious transfer with a memory-bounded receiver. In *Proceedings of 39th IEEE FOCS*, pages 493–502, 1998.
- [10] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.
- [11] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42th Annual IEEE Symposium on Foundations of Computer Science (FOCS '01)*, pages 136–145, 2001. Updated Version at <http://eprint.iacr.org/2000/067>.
- [12] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC '02)*, pages 494–503. ACM Press, 2002. Full version available at <http://eprint.iacr.org/2002/140>.
- [13] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.
- [14] C. Crépeau. Quantum oblivious transfer. *Journal of Modern Optics*, 41(12):2455–2466, 1994.
- [15] C. Crépeau, J. van de Graaf, and A. Tapp. Committed oblivious transfer and private multi-party computation. In *CRYPTO '95: Proceedings of the 15th Annual International Cryptology Conference on Advances in Cryptology*, pages 110–123. Springer-Verlag, 1995.
- [16] I. Damgård, S. Fehr, R. Renner, L. Salvail, and C. Schaffner. A tight high-order entropic uncertainty relation with applications in the bounded quantum-storage model. In *Advances in Cryptology — CRYPTO 2007*, 2007. Full version available at quant-ph/0612014.
- [17] I. Damgård, S. Fehr, L. Salvail, and C. Schaffner. Cryptography in the Bounded Quantum-Storage Model. In *Proceedings of 46th IEEE FOCS*, pages 449–458, 2005.
- [18] G. Estren. Universally composable committed oblivious transfer and multi-party computation assuming only basic black-box. M.Sc. thesis, School of Computer Science, McGill University, 2004.

- [19] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.
- [20] O. Goldreich and R. Vainish. How to solve any protocol problem - an efficiency improvement. In *Advances in Cryptology — CRYPTO '87*, Lecture Notes in Computer Science, pages 73–86. Springer-Verlag, 1988.
- [21] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
- [22] M. Hayashi. *Quantum Information: An introduction*. Springer, 2006.
- [23] J. Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC '88)*, pages 20–31. ACM Press, 1988.
- [24] A. Kitaev, D. Mayers, and J. Preskill. Superselection rules and quantum protocols. *Physical Review A*, 69:052326, 2004.
- [25] R. König, R. Renner, A. Bariska, and U. Maurer. Locking of accessible information and implications for the security of quantum cryptography. 2005. quant-ph/0512021.
- [26] H. K. Lo and H. F. Chau. Is quantum bit commitment really possible? *Physical Review Letters*, 78:3410–3413, 1997.
- [27] U. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.
- [28] U. Maurer and S. Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321. Springer-Verlag, 1997.
- [29] D. Mayers. Unconditionally secure quantum bit commitment is impossible. *Physical Review Letters*, 78:3414–3417, 1997.
- [30] S. Micali and P. Rogaway. Secure computation (abstract). In *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 392–404. Springer-Verlag, 1992.
- [31] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 4(2):151–158, 1991.
- [32] B. Pfitzmann and M. Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy (SP '01)*, page 184, 2001. Also available at <http://eprint.iacr.org/2000/066>.
- [33] M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981.
- [34] R. Renner. *Security of Quantum Key Distribution*. PhD thesis, ETH Zurich, Switzerland, 2005. Available at <http://arxiv.org/abs/quant-ph/0512258>.

- [35] R. Renner and R. König. Universally composable privacy amplification against quantum adversaries. In *Theory of Cryptography Conference — TCC '05*, volume 3378 of *Lecture Notes in Computer Science*, pages 407–425. Springer-Verlag, 2005. Also available at <http://arxiv.org/abs/quant-ph/0403133>.
- [36] R. Renner and S. Wolf. Simple and tight bounds for information reconciliation and privacy amplification. In *Advances in Cryptology — ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 199–216. Springer-Verlag, 2005.
- [37] L. Salvail. Quantum bit commitment from a physical assumption. In *Proceedings of CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 338–353, 1998.
- [38] A. Smith. Multi-party quantum computation. Masters Thesis, 2001. quant-ph/0111030.
- [39] R. Spekkens and T. Rudolph. Degrees of concealment and bindingness in quantum bit commitment protocols. *Physical Review A*, 65(012310), 2002.
- [40] D. Unruh. Formal security in quantum cryptology. Student research project, Institut für Algorithmen und Kognitive Systeme, University of Karlsruhe, 2002.
- [41] D. Unruh. Simulatable security for quantum protocols. quant-ph/0409125, 2004.
- [42] J. van de Graaf. Towards a formal definition of security for quantum protocols. Ph.D. thesis, Dpartement d'informatique et de r.o., Universit de Montral, 1998. <http://www.cs.mcgill.ca/~crepeau/PS/these-jeroen.ps>.
- [43] J. Watrous. Zero-knowledge against quantum attacks. quant-ph/0511020, 2005.
- [44] S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.
- [45] J. Wullschleger. Oblivious-transfer amplification. In *Advances in Cryptology — EUROCRYPT '07*, Lecture Notes in Computer Science. Springer-Verlag, 2007.
- [46] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS '82)*, pages 160–164, 1982.

## A Proof of Lemma 2.3

To prove Lemma 2.3, we need Lemma A.2 below and the following Theorem A.1, which is Corollary 3.4 in the full version of [16].

**Theorem A.1** (Uncertainty Relation [16]). *Let  $\rho \in \mathbb{S}(\mathcal{H}_2^{\otimes n})$  be an arbitrary quantum state. Let  $\Theta = (\Theta_1, \dots, \Theta_n)$  be uniformly distributed over  $\{+, \times\}$  and let  $X = (X_1, \dots, X_n)$  be the outcome when measuring  $\rho$  in basis  $\Theta$ . Then for any  $0 < \lambda < \frac{1}{2}$*

$$H_{\min}^\varepsilon(X|\Theta) \geq \left(\frac{1}{2} - 2\lambda\right)n$$

with  $\varepsilon = \exp\left(-\frac{\lambda^2 n}{32(2-\log(\lambda))^2}\right)$ .

**Lemma A.2.** *For all  $0 < x \leq 0.5$ , we have*

$$\frac{1}{(2 - \log(x))^2} \geq \frac{e^3 \ln(2)^2}{54} \cdot x .$$

*Proof.* Since  $(2 - \log(x))^{-2} \rightarrow 0$  for  $x \rightarrow 0$ , it suffices to show that, for all  $0 < x \leq 0.5$ ,

$$\frac{d}{dx} \frac{1}{(2 - \log(x))^2} = \frac{2}{\ln(2)} \frac{1}{(2 - \log(x))^3 x} \geq \frac{e^3 \ln(2)^2}{54} ,$$

which is equivalent to require that

$$f(x) := \frac{\ln(2)}{2} (2 - \log(x))^3 x \leq \frac{54}{e^3 \ln(2)^2} .$$

We have

$$f'(x) = -3(2 - \log(x))^2 + (2 - \log(x))^3 \ln(2) ,$$

and since the polynomial  $-3x^2 + x^3 \ln(2)$  has a double root at 0 and a single root at  $3/\ln(2)$ , and is positive if and only if  $x > 3/\ln(2)$ , it follows that  $f(x)$  has one double root at 4, and one single root at  $4/e^3$ . It is positive for  $0 < x < 4/e^3$  and negative for  $4/e^3 < x \leq 0.5$ . Hence,  $f(x)$  is maximal for  $x = 4/e^3$ , where  $f(4/e^3) = 54/(e^3 \ln(2)^2)$ .  $\square$

*Proof of Lemma 2.3.* Following the standard approach (see also [16]), we consider a purified version of our situation: Alice creates  $n$  EPR pairs, and sends the second half of each pair to Bob. His measurement is then applied onto the second half of these pairs, which has output  $K$ . Then, we choose uniform a random basis  $\Theta \in \{+, \times\}^n$ , and measure the first half in this basis, which gives us the string  $X$ . The output of the purified situation is identical to the situation in the statement, however it allows us to apply Corollary A.1.

From  $10\sqrt[3]{n^2 \log(1/\varepsilon)} = n/2\sqrt[3]{8000 \log(1/\varepsilon)/n}$  follows that  $n/2 > 10\sqrt[3]{n^2 \log(1/\varepsilon)}$  if and only if  $n > 8000 \log(1/\varepsilon)$ . Thus, nothing has to be shown if  $n \leq 8000 \log(1/\varepsilon)$ . If  $n > 8000 \log(1/\varepsilon)$ , we choose  $\lambda := 5\sqrt[3]{1/n \cdot \log(1/\varepsilon)}$  and  $\lambda' := 1/n \cdot \log(1/\varepsilon)$ , and get

$$\lambda = 5\sqrt[3]{\frac{1}{n} \cdot \log(1/\varepsilon)} < 5\sqrt[3]{\frac{1}{20^3 \cdot \log(1/\varepsilon)} \cdot \log(1/\varepsilon)} = 5\sqrt[3]{\frac{1}{20^3}} = \frac{1}{4} .$$

The statement follows from

$$\exp\left(-\frac{\lambda^2 n}{32 \cdot (2 - \log(\lambda))^2}\right) \leq \exp\left(-\frac{\lambda^3 n}{180}\right) = 2^{-\frac{\log(e)}{180} \lambda^3 n} \leq 2^{-\frac{1}{5^3} \lambda^3 n} \leq 2^{-\log(1/\varepsilon)} = \varepsilon .$$

$\square$

## B Oblivious Transfer from ROT

Oblivious transfer is defined as follows:

**Definition B.1** (Oblivious transfer). *The functionality  $\binom{2}{1}\text{-OT}^\ell$  receives input  $(x_0, x_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  from  $A$  and  $c \in \{0, 1\}$  from  $B$ , and sends  $y := x_c$  to  $B$ .*

The protocol OTfromROT, proposed in [5], securely implements OT using ROT and Comm.

**Protocol 3: OTfromROT<sub>A</sub>**

- 1:** Receive input  $(x_0, x_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  and  $(x'_0, x'_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  from ROT.
- 2:** Receive  $d \in \{0, 1\}$  from Comm.
- 3:** Send  $(m_0, m_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  to Comm, where  $m_i := x_i \oplus x'_{i \oplus d}$ .

**Protocol 4: OTfromROT<sub>B</sub>**

- 1:** Receive input  $c \in \{0, 1\}$  and  $(c', y') \in \{0, 1\} \times \{0, 1\}^\ell$  from ROT.
- 2:** Send  $d := c' \oplus c$  to Comm.
- 3:** Receive  $(m_0, m_1) \in \{0, 1\}^\ell \times \{0, 1\}^\ell$  from Comm and output  $y := m_c \oplus y'$  to B.

**Theorem B.2.** *For every  $m > 0$ ,  $OTfromROT((^2_1)\text{-}ROT^\ell \| Comm)$  implements  $(^2_1)\text{-}OT^\ell$  with no error, secure against  $m$ -bounded adversaries using  $m$ -bounded simulators.*

*Proof.* It is easy to verify that the protocol is correct, if  $\mathbb{A} = \emptyset$ .

Let  $\mathbb{A} = \{\mathbf{A}\}$  and  $\mathbf{A}_A$  be a quantum adversary.  $\mathbf{A}_A$  receives some auxiliary input<sup>7</sup>, and outputs  $(x'_0, x'_1)$  which are the inputs to ROT<sub>A</sub>. Then it receives  $d$ , and finally output  $(m_0, m_1)$  and some auxiliary output. The simulator  $\mathbf{S}_A$  works as follows. It receives some auxiliary input, and then executes  $\mathbf{A}_A$ , using the auxiliary input. It stores the values  $(x'_0, x'_1)$  returned by  $\mathbf{A}_A$ , and sends it a value  $d$  chosen uniformly at random.  $\mathbf{A}_A$  then outputs  $(m_0, m_1)$  and some auxiliary output. The simulator outputs the auxiliary output and sends the values  $x_i := m_i \oplus x'_{i \oplus d}$  for  $i \in \{0, 1\}$  to OT. It is easy to verify that the real and the simulated situations give exactly the same output distribution.

Let  $\mathbb{A} = \{\mathbf{B}\}$  and  $\mathbf{A}_B$  be a quantum adversary.  $\mathbf{A}_B$  receives some auxiliary input and outputs  $(c'_0, y')$ , which are the inputs to ROT<sub>A</sub>, and a value  $d$ . Then it receives the values  $(m_0, m_1)$ , and outputs some auxiliary output. The simulator works as follows. It receives some auxiliary input, and then executes  $\mathbf{A}_B$  on the auxiliary input, which outputs  $(c'_0, y')$  and a value  $d$ . The simulator now sends  $c := c' \oplus d$  to OT, and receives a value  $y$  back. Then, it sets  $m_{c' \oplus d} := y \oplus y'$ , chooses the other value  $m_{c' \oplus d \oplus 1}$  uniformly at random, and sends  $(m_0, m_1)$  to  $\mathbf{A}_B$ . Finally, it outputs the auxiliary output that  $\mathbf{A}_B$  returns. It is easy to verify that the real and the simulated situations give exactly the same output distribution.  $\square$

## C Bit-Commitment from ROT

In [17], a bit-commitment protocol is presented and proved secure for a weak binding condition. In [16], it is shown that the same protocol is in fact also secure under a stronger binding condition. However, as for ROT, their proof does not take auxiliary inputs into account. In a similar way

---

<sup>7</sup>Now, auxiliary inputs and output are always both classical and quantum.

as ROT, their protocol could be proven secure in our framework. But because protocols can be composed in our framework, we can now give a much simpler proof: We can implement bit-commitment based directly on ROT. The composition theorem then implies that if ROT is replaced by an instance of BQS-OT, the bit-commitment protocol remains secure. The BC functionality is defined as follows.

**Definition C.1.** *The functionality BC has two phases, which are defined as follows:*

- *Commit:* BC receives  $b \in \{0, 1\}$  from A and sends  $\perp$  to B.
- *Open:* BC receives  $a \in \{0, 1\}$  from A. If  $a = 1$ , it sends  $b$  to B. Otherwise, it sends  $\perp$ .

Let  $\binom{2}{1}\text{-TOR}^\ell$  be a reversed version of  $\binom{2}{1}\text{-ROT}^\ell$ , i.e., B is the sender and A is the receiver. The protocol OTtoBC = (OTtoBC<sub>A</sub>, OTtoBC<sub>B</sub>) uses  $\binom{2}{1}\text{-TOR}^\ell$  and a noiseless unidirectional classical Comm from A to B to implement BC. We now first describe the actions of the committer.

**Protocol 5: OTtoBC<sub>A</sub>**

**Commit:**

1. Receive input  $b$  from A and  $(c, y) \in \{0, 1\} \times \{0, 1\}^n$  from TOR.
2. Send  $m := b \oplus c$  to Comm.

**Open:**

1. Receive input  $a$  from A. If  $a = 1$ , then send  $(b, y)$  to Comm, and  $(\perp, \perp)$  otherwise.

The actions of the verifier are specified by:

**Protocol 6: OTtoBC<sub>B</sub>**

**Commit:**

1. Receive  $(x_0, x_1)$  from TOR .
2. Receive  $m$  from Comm and output  $\perp$ .

**Open:**

1. Receive  $(b, y)$  from Comm.
2. If  $(b, y) \neq (\perp, \perp)$  and  $x_{b \oplus m} = y$ , then output  $b$ , and  $\perp$  otherwise.

**Theorem C.2.** *For every  $m > 0$ , OTtoBC( $\binom{2}{1}\text{-TOR}^\ell \parallel \text{Comm}$ ) implements BC with an error of at most  $2^{-\ell}$ , secure against  $m$ -bounded adversaries using  $m$ -bounded simulators.*

*Proof.* It is easy to verify that the protocol is correct, if  $\mathbb{A} = \emptyset$

Let  $\mathbb{A} = \{\mathbb{A}\}$  and  $\mathbb{A}_A$  be a quantum adversary. In the commit phase, it receives some auxiliary input, sends  $(c, y)$  to TOR and  $m$  to Comm and outputs some auxiliary output. In the open phase, it receives some auxiliary input, sends  $(b, y')$  to Comm, and outputs some auxiliary output. Note

that in the real execution, if  $(b, y') = (c \oplus m, y)$ , the protocol outputs  $b$  to  $\mathbf{B}$  in the open phase. On the other hand, if  $b \neq c \oplus m$ , the protocol will only output  $b$  to  $\mathbf{B}$ , if  $y = x_{1-c}$ . Since  $x_{1-c}$  is chosen uniformly at random, this will only happen with a probability of  $2^{-\ell}$ . The simulator  $\mathbf{S}_A$  does the following: In the commit phase, it receives the auxiliary input, and sends it to  $\mathbf{A}_A$  to run the commit phase, from which it receives  $(c, y)$ ,  $m$ , and some auxiliary output. It outputs the auxiliary output, sends  $c \oplus m$  to  $\mathbf{BC}$ , and saves  $c \oplus m$  in his classical memory. In the open phase, it receives some auxiliary input and sends it to  $\mathbf{A}_A$  to run the open phase. It receives  $(b, y')$  from  $\mathbf{A}_A$ , and sends  $a = 1$  to  $\mathbf{BC}$  if  $b = c \oplus m$ , and  $a = 0$  otherwise. Finally, it outputs the auxiliary output of  $\mathbf{A}_A$ . It is easy to verify that simulation is equal to the real execution, except with probability at most  $2^{-\ell}$ .

Let  $\mathbb{A} = \{\mathbf{B}\}$  and  $\mathbf{A}_B$  be a quantum adversary. In the commit phase, it receives some auxiliary input, sends  $(x_0, x_1)$  to  $\text{ROT}$ , receives  $m$ , and outputs some auxiliary output. In the open phase, it receives some auxiliary input and  $(b, y)$  from  $\text{Comm}$ , and outputs some auxiliary output. The simulator  $\mathbf{S}_B$  does the following: In the commit phase, it receives some auxiliary input, and sends it to  $\mathbf{A}_B$ , from which it receives  $(x_0, x_1)$ . Then, it sends a value  $m$  chosen uniformly at random to  $\mathbf{A}_B$ , and gets some auxiliary output back. It outputs the auxiliary output, and stores  $(x_0, x_1)$  and  $m$  in the classical memory. In the open phase, it receives some auxiliary input, and a value  $b'$  from  $\mathbf{BC}$ . If  $b' = \perp$ , it sets  $y := \perp$ , and  $y := x_{b' \oplus m}$  otherwise. It sends the auxiliary input and  $(b', y)$  to  $\mathbf{A}_B$ , and outputs the auxiliary output returned by  $\mathbf{A}_B$ . It is easy to see that the simulation produces always exactly the same output as the simulation.  $\square$

Let  $\text{BQS-TO}$  be the same protocol as  $\text{BQS-OT}$ , but in the opposite direction. Using Theorem 4.7 and C.2, as well as Theorem 3.2, and by choosing  $\ell := \log 1/\varepsilon$ , we get

**Theorem C.3.** *The Protocol  $OTtoBC(\text{BQS-TO}(Q\text{-}\mathbf{Comm})\|\mathbf{Comm})\|\mathbf{Comm}$  implements  $\mathbf{BC}$  with an error of at most  $6\varepsilon$ , secure against  $m$ -bounded adversaries using  $m$ -bounded simulators, if*

$$10m \leq n - 20\sqrt[3]{n^2 \log \frac{1}{\varepsilon}} - 20 \log \frac{1}{\varepsilon} - 4$$